A Sampling-Based Approach to Spacecraft Autonomous Maneuvering with Safety Specifications

Joseph A. Starek*, Brent W. Barbee†, Marco Pavone*

*Autonomous Systems Lab
Dept. of Aeronautics & Astronautics
Stanford University

†Navigation and Mission Design Branch
Goddard Space Flight Center
NASA

AAS-GNC 2015
February 3rd, 2015

Outline

- Autonomous Vehicle Safety

Outline

- Autonomous Vehicle Safety
- Spacecraft Safety

Outline

- Autonomous Vehicle Safety
- Spacecraft Safety
- Safety in CWH Dynamics

Outline

- Autonomous Vehicle Safety
- Spacecraft Safety
- Safety in CWH Dynamics
- Numerical Experiments

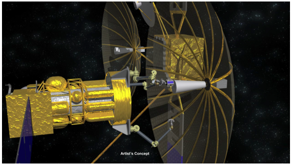
Outline

- Autonomous Vehicle Safety
- Spacecraft Safety
- Safety in CWH Dynamics
- Numerical Experiments
- Conclusions and Future Work

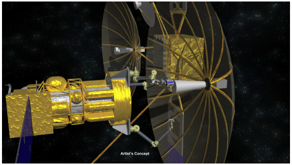
The Need for Safe Autonomy

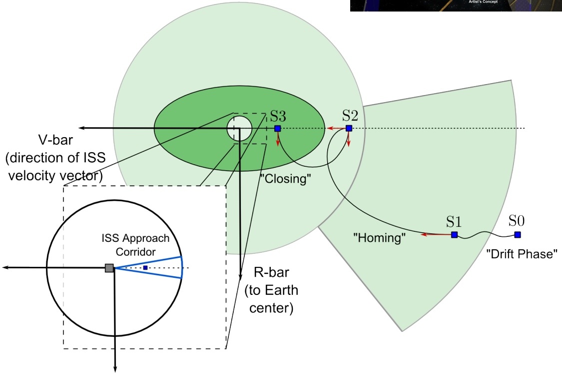
The Need for Safe Autonomy

- Satellite servicing (DARPA Phoenix Mission)



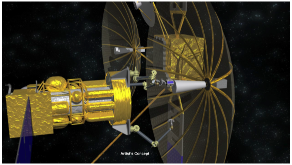
- Satellite servicing (DARPA Phoenix Mission)
- Automated rendezvous





The Need for Safe Autonomy

- Satellite servicing (DARPA Phoenix Mission)
- Automated rendezvous



Key Question

How do we implement a general, automated spacecraft planning framework with hard safety specifications?

Original Contribution

Our work:

1. Establishes a **provably-correct framework** for the *systematic* encoding of safety specifications into the spacecraft trajectory generation process
2. Derives an efficient **one-burn escape maneuver policy** for proximity operations near circular orbit

Previous Work

Spacecraft rendezvous approaches with explicit characterizations of safety:

- Kinematic path optimization *[Jacobsen, Lee, et al., 2002]*
- Artificial potential functions *[Roger and McInnes, 2000]*
- MILP formulations *[Breger and How, 2008]*
- Safety ellipses *[Gaylor and Barbee, 2007] [Naasz, 2005]*
- Motion planning *[Frazzoli, 2003]*
- Robust Model-Predictive Control *[Carson, Açikmeșe, et al., 2008]*
- Forced equilibria *[Weiss, Baldwin, et al., 2013]*

Types of Spacecraft Rendezvous Safety

- **Passive Trajectory Protection**: Constrain coasting trajectories to avoid collisions up to a given horizon time
- **Active Trajectory Protection**: Implement an *actuated* escape maneuver to save/abort a mission

Design Choice

We emphasize *active safety* as it is the less-conservative approach

Vehicle Trajectory Safety

Definition (Trajectory Safety Problem)

For all possible failure times $t_{\text{fail}} \in \mathcal{T}_{\text{fail}}$ and failure modes $\mathcal{U}_{\text{fail}}(\mathbf{x}(t_{\text{fail}}))$, we seek a sequence of admissible actions $\mathbf{u}(\tau) \in \mathcal{U}_{\text{fail}}(\mathbf{x}(t_{\text{fail}}))$ from $\mathbf{x}(t_{\text{fail}})$ such that the remaining trajectory is safe.

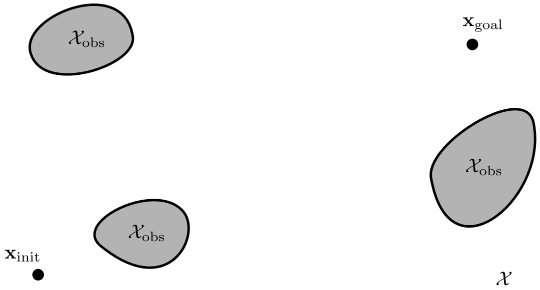
Examples:

- **Rovers/Land vehicles**: Come to a complete stop
- **Manipulators**: Return to previous configuration, disengage, or execute emergency plan
- **UAV's**: Enter a safe loiter pattern
- **Spacecraft**: Less straightforward; generally require mission-specific solutions (with human oversight)

Vehicle Trajectory Safety

Definition (Trajectory Safety Problem)

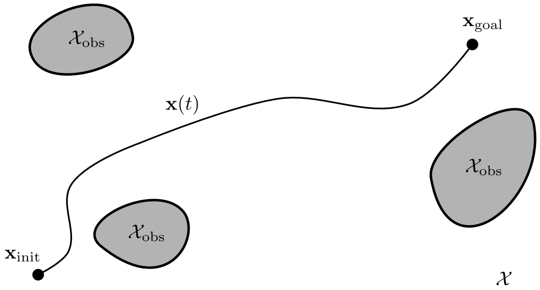
For all possible failure times $t_{\text{fail}} \in \mathcal{T}_{\text{fail}}$ and failure modes $\mathcal{U}_{\text{fail}}(\mathbf{x}(t_{\text{fail}}))$, we seek a sequence of admissible actions $\mathbf{u}(\tau) \in \mathcal{U}_{\text{fail}}(\mathbf{x}(t_{\text{fail}}))$ from $\mathbf{x}(t_{\text{fail}})$ such that the remaining trajectory is safe.



Vehicle Trajectory Safety

Definition (Trajectory Safety Problem)

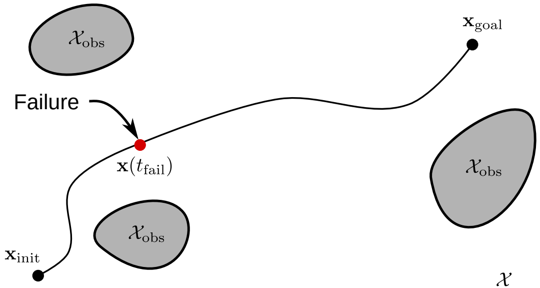
For all possible failure times $t_{\text{fail}} \in \mathcal{T}_{\text{fail}}$ and failure modes $\mathcal{U}_{\text{fail}}(\mathbf{x}(t_{\text{fail}}))$, we seek a sequence of admissible actions $\mathbf{u}(\tau) \in \mathcal{U}_{\text{fail}}(\mathbf{x}(t_{\text{fail}}))$ from $\mathbf{x}(t_{\text{fail}})$ such that the remaining trajectory is safe.



Vehicle Trajectory Safety

Definition (Trajectory Safety Problem)

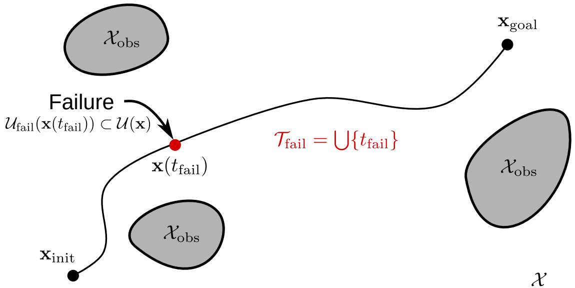
For all possible failure times $t_{\text{fail}} \in \mathcal{T}_{\text{fail}}$ and failure modes $\mathcal{U}_{\text{fail}}(\mathbf{x}(t_{\text{fail}}))$, we seek a sequence of admissible actions $\mathbf{u}(\tau) \in \mathcal{U}_{\text{fail}}(\mathbf{x}(t_{\text{fail}}))$ from $\mathbf{x}(t_{\text{fail}})$ such that the remaining trajectory is safe.



Vehicle Trajectory Safety

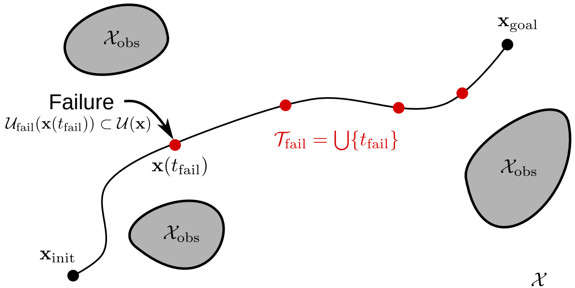
Definition (Trajectory Safety Problem)

For all possible failure times $t_{\text{fail}} \in \mathcal{T}_{\text{fail}}$ and failure modes $\mathcal{U}_{\text{fail}}(\mathbf{x}(t_{\text{fail}}))$, we seek a sequence of admissible actions $\mathbf{u}(\tau) \in \mathcal{U}_{\text{fail}}(\mathbf{x}(t_{\text{fail}}))$ from $\mathbf{x}(t_{\text{fail}})$ such that the remaining trajectory is safe.



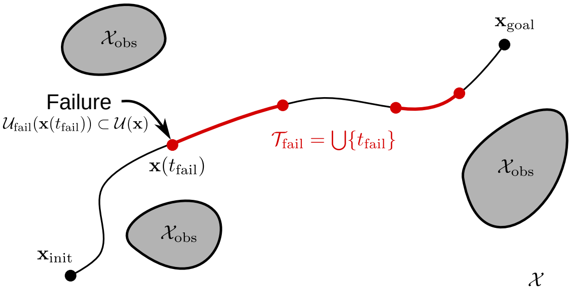
Definition (Trajectory Safety Problem)

For all possible failure times $t_{\text{fail}} \in \mathcal{T}_{\text{fail}}$ and failure modes $\mathcal{U}_{\text{fail}}(\mathbf{x}(t_{\text{fail}}))$, we seek a sequence of admissible actions $\mathbf{u}(\tau) \in \mathcal{U}_{\text{fail}}(\mathbf{x}(t_{\text{fail}}))$ from $\mathbf{x}(t_{\text{fail}})$ such that the remaining trajectory is safe.



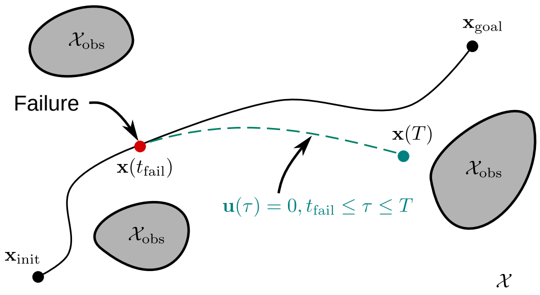
Definition (Trajectory Safety Problem)

For all possible failure times $t_{\text{fail}} \in \mathcal{T}_{\text{fail}}$ and failure modes $\mathcal{U}_{\text{fail}}(\mathbf{x}(t_{\text{fail}}))$, we seek a sequence of admissible actions $\mathbf{u}(\tau) \in \mathcal{U}_{\text{fail}}(\mathbf{x}(t_{\text{fail}}))$ from $\mathbf{x}(t_{\text{fail}})$ such that the remaining trajectory is safe.



Challenge: Infinite-Horizon Safety

Finite-horizon safety guarantees can ultimately violate constraints:

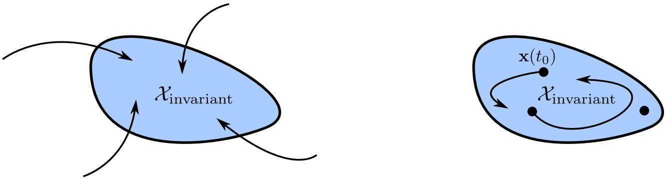


Idea: Positively-Invariant Sets

Definition (Positively-Invariant Set)

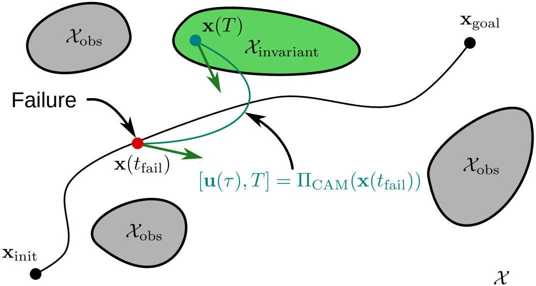
A set $\mathcal{X}_{\text{invariant}}$ is positively invariant with respect to $\dot{\mathbf{x}} = f(\mathbf{x})$ if and only if

$$\mathbf{x}(t_0) \in \mathcal{X}_{\text{invariant}} \implies \mathbf{x}(t) \in \mathcal{X}_{\text{invariant}},\, t \geq t_0$$



Definition (Vehicle State Safety)

A state is *safe* if and only if there exists, under all failure conditions, a safe, dynamically-feasible trajectory that *navigates the vehicle to a safe, stable positively-invariant set*.



Finite-Time Trajectory Safety

$$
\begin{aligned}
\underset{t_f, \mathbf{x}(t), \mathbf{u}(t)}{\text{minimize}} \quad & J(\mathbf{x}, \mathbf{u}, t) \\
\text{subject to} \quad & \dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t), t) && \text{(Dynamics)} \\
& \mathbf{x}(t_0) = \mathbf{x}_0 && \text{(Initial Condition)} \\
& \mathbf{x}(t_f) \in \mathcal{X}_{\text{invariant}} && \text{(Invariant Termination)} \\
& \mathbf{u}(t) \in \mathcal{U}_{\text{fail}}(\mathbf{x}_0) && \text{(Control Admissibility)} \\
& g_i(\mathbf{x}, \mathbf{u}) \le 0, i = [1, \ldots, p] && \text{(Inequality Constraints)} \\
& h_j(\mathbf{x}, \mathbf{u}) = 0, j = [1, \ldots, q] && \text{(Equality Constraints)}
\end{aligned}
$$

Challenge: Solving the Finite-Time Safety Problem under Failures

For a K-fault tolerant spacecraft with N control components (thrusters, momentum wheels, CMG's, etc), this yields:

$$N_{\text{fail}} = \sum_{k=0}^{K} \binom{N}{k} = \sum_{k=0}^{K} \frac{N!}{k!(N-k)!}$$

total optimization problems (one for each $\mathcal{U}_{\text{fail}}$) for each failure time t_{fail}.

Idea: Simplify the Finite-Time Safety Problem

Theorem (Sufficient Fault-Tolerant Active Safety)

1. *From each* $\mathbf{x}(t_{fail})$*, prescribe a Collision-Avoidance Maneuver* $\Pi_{CAM}(\mathbf{x})$ *that gives a horizon* T *and escape sequence* $\mathbf{u}$ *that satisfies* $\mathbf{x}(T) \in \mathcal{X}_{invariant}$ *and* $\mathbf{u}(\tau) \subset \mathcal{U}$ *for all* $t_{fail} \leq \tau \leq T$.
2. *For each failure mode* $\mathcal{U}_{fail}(\mathbf{x}(t_{fail})) \subset \mathcal{U}(\mathbf{x}(t_{fail}))$ *up to tolerance* K*, check if* $\mathbf{u} = \Pi_{CAM}(\mathbf{x}) \subset \mathcal{U}_{fail}$.

Idea: Simplify the Finite-Time Safety Problem

Theorem (Sufficient Fault-Tolerant Active Safety)

1. *From each $\mathbf{x}(t_{fail})$, prescribe a Collision-Avoidance Maneuver $\Pi_{CAM}(\mathbf{x})$ that gives a horizon T and escape sequence $\mathbf{u}$ that satisfies $\mathbf{x}(T) \in \mathcal{X}_{invariant}$ and $\mathbf{u}(\tau) \subset \mathcal{U}$ for all $t_{fail} \leq \tau \leq T$.*
2. *For each failure mode $\mathcal{U}_{fail}(\mathbf{x}(t_{fail})) \subset \mathcal{U}(\mathbf{x}(t_{fail}))$ up to tolerance K, check if $\mathbf{u} = \Pi_{CAM}(\mathbf{x}) \subset \mathcal{U}_{fail}$.*

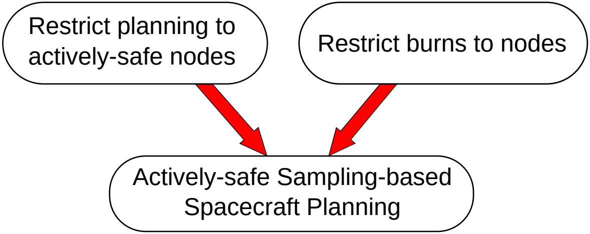
Key Simplifications

Removes decision variables $\mathbf{u}$, reducing to:

- a test of escape control feasibility under failure(s)
- numerical integration for satisfaction of dynamics
- an *a posteriori* check of constraints g_i and h_j

Safe Sampling-Based Spacecraft Planning

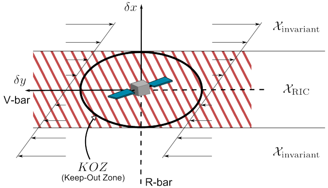
Solution is in exact form required for sampling-based motion planning.

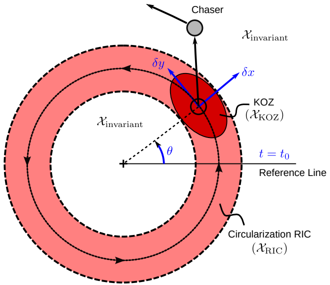


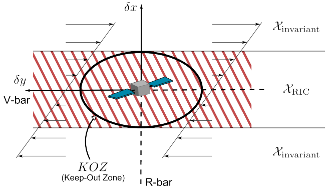
Incorporating Safety Constraints:

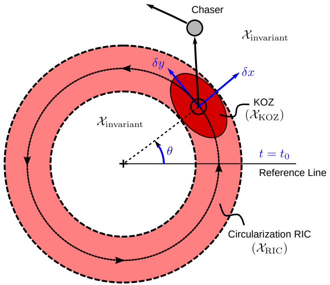
- Add CAM policy generation to sampling algorithm
- Include CAM-trajectory collision-checking in tests of sample feasibility

Example: CAM Policy Design
Using CWH Set Invariance for CAMs









Circular Clohessy-Wiltshire-Hill (CWH) CAM policy:

1. Coast from $\mathbf{x}(t)$ to some new $T > t$ such that $\mathbf{x}(T^-)$ lies at a position in $\mathcal{X}_{\text{invariant}}$.
2. Circularize the orbit at $\mathbf{x}(T)$ such that $\mathbf{x}(T^+) \in \mathcal{X}_{\text{invariant}}$
3. Coast along the new orbit (horizontal drift along the in-track axis) in $\mathcal{X}_{\text{invariant}}$

Example: CAM Policy Design

Choosing the Circularization Time, T

CWH Finite-Time Safety Problem:

$$
\begin{aligned}
\text{Given:} \quad & \mathbf{x}(t), \mathbf{u}(\tau) = \mathbf{0}, t \leq \tau < T \\
\underset{T}{\text{minimize}} \quad & \Delta v_{\text{circ}}^2(T) \\
\text{subject to} \quad & \dot{\mathbf{x}}(\tau) = f(\mathbf{x}(\tau), \mathbf{0}, \tau) && \text{(Dynamics)} \\
& \mathbf{x}(\tau) \notin \mathcal{X}_{\text{KOZ}} && \text{(KOZ Avoidance)} \\
& \mathbf{x}(T^+) \in \mathcal{X}_{\text{invariant}} && \text{(Invariant Termination)}
\end{aligned}
$$

Key Result

Can be reduced to an analytical expression that is solvable in milliseconds

Scenario

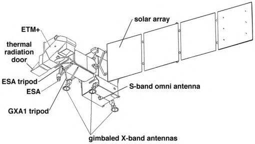
- Simulates an automated approach to LandSat-7 (*e.g.*, for servicing) between pre-specified waypoints
- Calls on the Fast Marching Tree (FMT*) algorithm for implementation

Scenario

- Simulates an automated approach to LandSat-7 (*e.g.*, for servicing) between pre-specified waypoints
- Calls on the Fast Marching Tree (FMT*) algorithm for implementation

Assumptions:

- Begins at insertion into a coplanar circular orbit sufficiently close to the target
- The target is nadir-pointing
- The chaser is nominally nadir-pointing, or executes a "turn-burn-turn" along CAMs

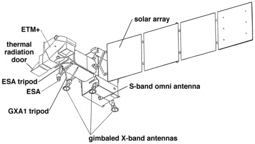


Scenario

- Simulates an automated approach to LandSat-7 (*e.g.*, for servicing) between pre-specified waypoints
- Calls on the Fast Marching Tree (FMT*) algorithm for implementation

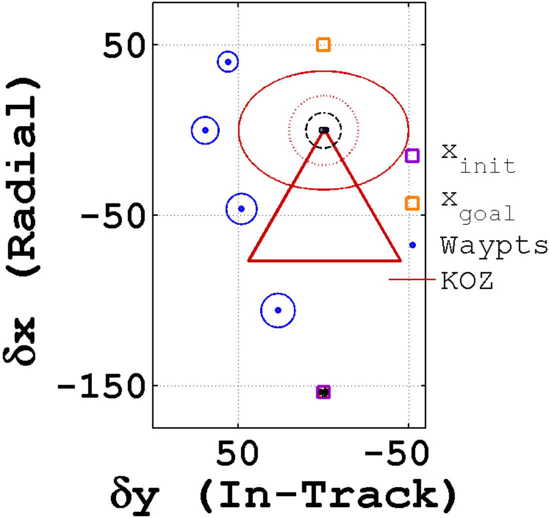
Constraints:

- **Plume impingement:** No exhaust plume impingement
- **Collision avoidance:** Clearance of an elliptic Keep-Out Zone (KOZ)
- **Target communication:** Target comm lobe avoidance
- **Safety:** Two-fault tolerance to stuck-off failures

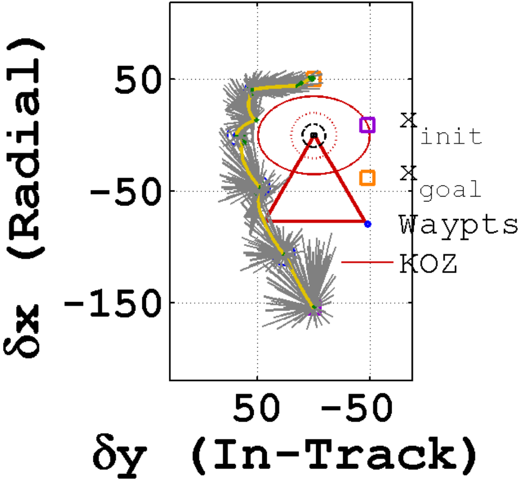


Motion Planning Problem

Motion planning query:

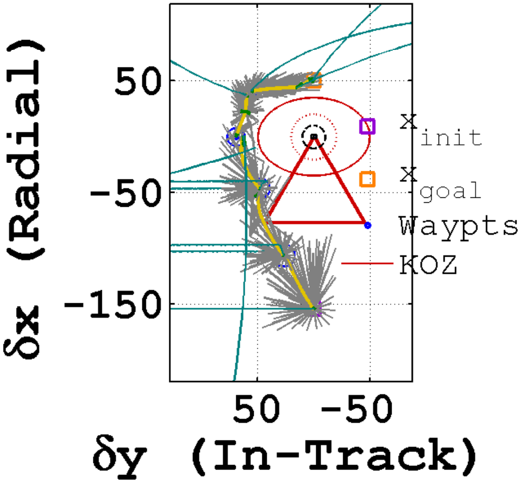


Motion planning solutions:



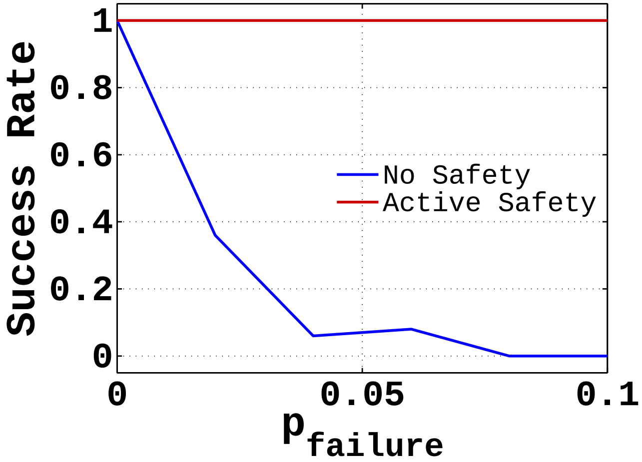
Motion Plan Comparison

Motion planning solutions:

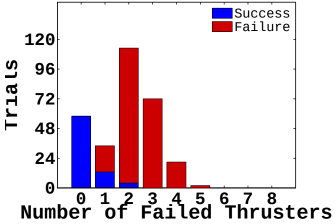


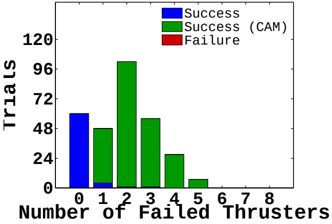
Success Rate Comparison

Success comparison as a function of thruster failure probability, computed over 50 trials:



Success comparison as a function of thruster failure probability, computed over 50 trials:





Conclusions

Key Ideas

1. Use termination constraints inside safe, stable, positively-invariant sets for infinite-horizon maneuver safety
2. Embed invariant-set constraints into sampling-based algorithms for safety-constrained planning

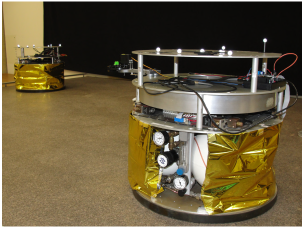
Synopsis

- Demonstrated the idea for failure-tolerant circular CWH planning
- CAM policies can be precomputed offline for more efficient online computation

Future Work

Future Goals

- Extend to thruster stuck-on and mis-allocation failures
- Account for localization uncertainty
- Apply these notions to small-body proximity operations



Thank you!

Joseph A. Starek, Brent W. Barbee, and
Marco Pavone

Aeronautics & Astronautics — Stanford University
Navigation and Mission Design — NASA GSFC

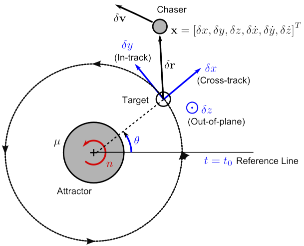
jstarek@stanford.edu

Clohessy-Wiltshire-Hill (CWH) Equations

- Motion is linearized about a moving reference point in circular orbit:

$$\mathbf{x} = [\delta x, \delta y, \delta z, \delta \dot{x}, \delta \dot{y}, \delta \dot{z}]^{\mathrm{T}}$$

$$\mathbf{u} = \frac{1}{m}[F_x, F_y, F_z]^{\mathrm{T}}$$



- Yields LTI dynamics:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$$

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 3{n_{\mathrm{ref}}}^2 & 0 & 0 & 0 & 2n_{\mathrm{ref}} & 0 \\ 0 & 0 & 0 & -2n_{\mathrm{ref}} & 0 & 0 \\ 0 & 0 & -{n_{\mathrm{ref}}}^2 & 0 & 0 & 0 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Generalized Mover's Problem

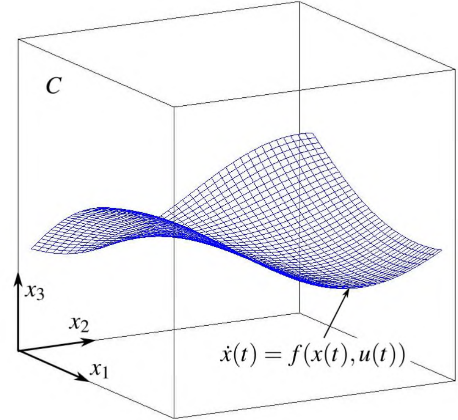
Definition (Optimal Motion Planning Problem)

Given $\mathcal{X}$, $\mathcal{X}_{\text{obs}}$, $\mathcal{X}_{\text{free}}$, and J, find an action trajectory $\mathbf{u} : [0,\ T] \rightarrow \mathcal{U}$ yielding a feasible path $\mathbf{x}(t) \in \mathcal{X}_{\text{free}}$ over *time horizon* $t \in [0,\ T]$, which reaches the *goal region* $\mathbf{x}(T) \in \mathcal{X}_{\text{goal}}$ and *minimizes* the cost functional $J = \int_0^T c(\mathbf{x}(t), \mathbf{u}(t))\, \mathrm{d}t$.

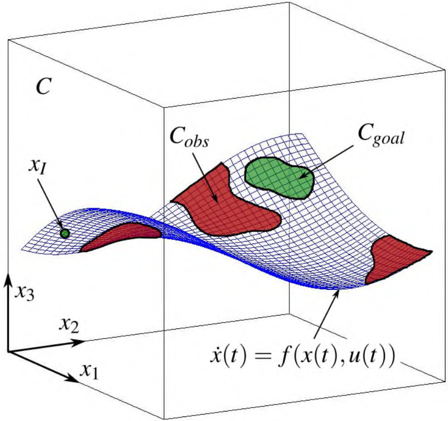
Characteristics:

- PSPACE-hard (and therefore NP-hard)
- Requires kinodynamic motion planning
- Almost certainly requires approximate algorithms, tailored to the particular application

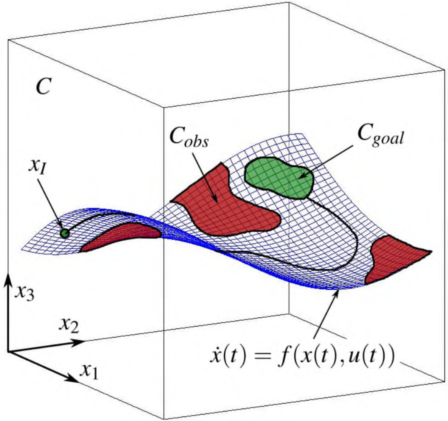
Generalized Mover's Problem



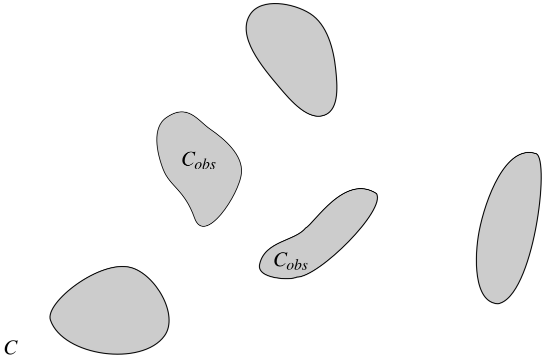
Generalized Mover's Problem



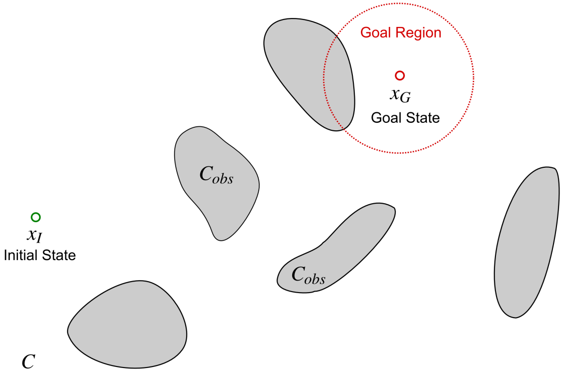
Generalized Mover's Problem



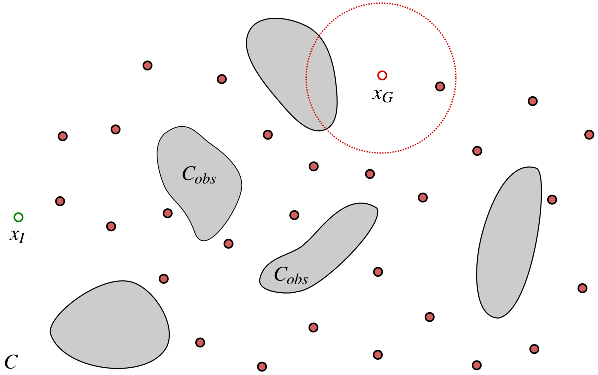
The FMT* Algorithm



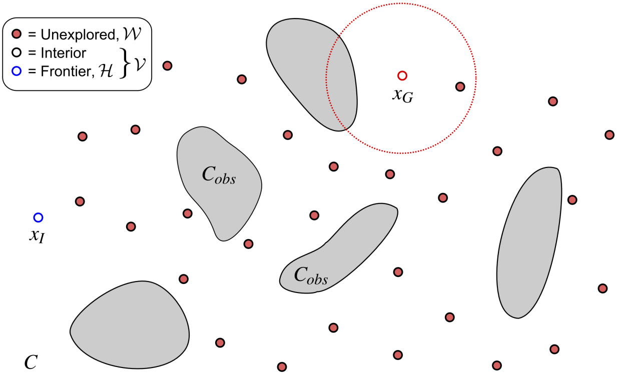
The FMT* Algorithm



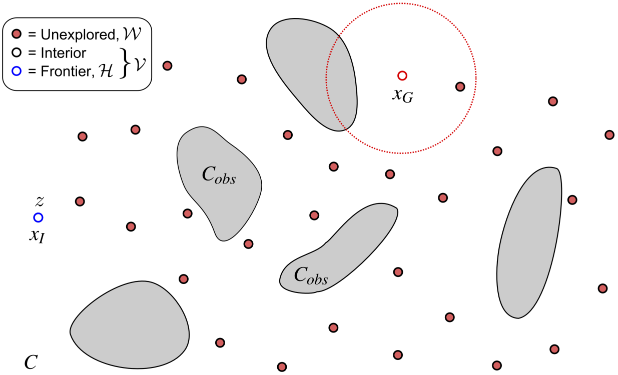
The FMT* Algorithm



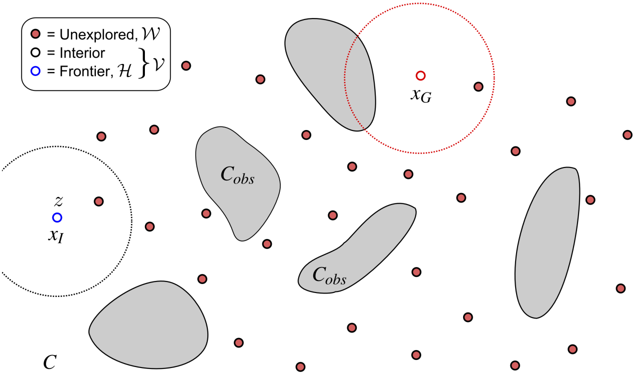
The FMT* Algorithm



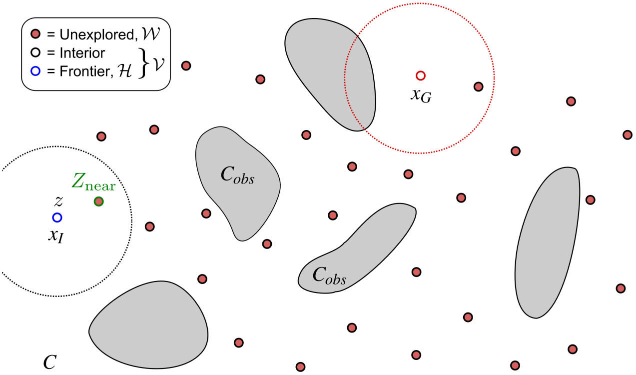
The FMT* Algorithm



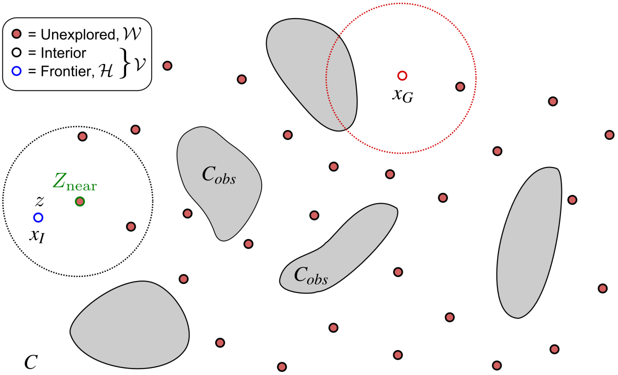
The FMT* Algorithm



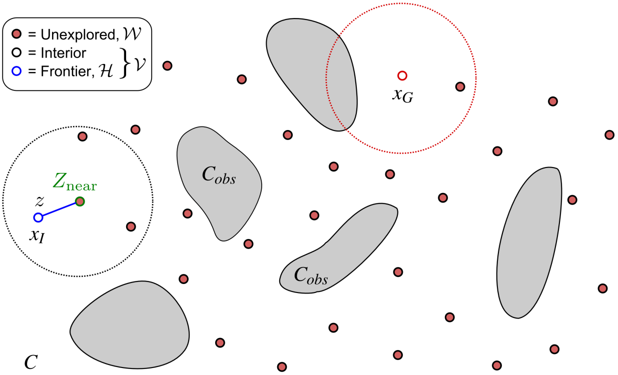
The FMT* Algorithm



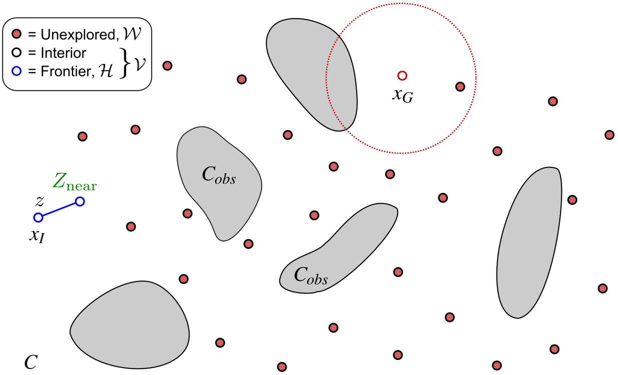
The FMT* Algorithm



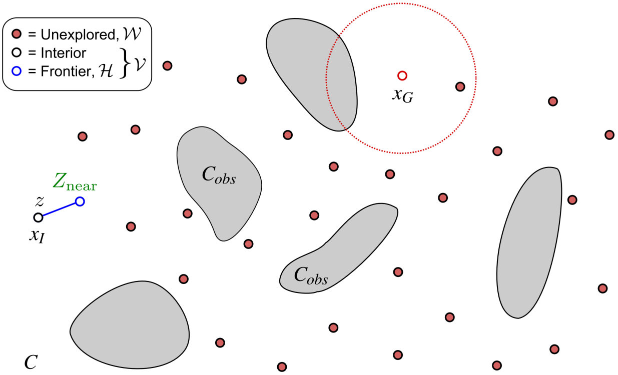
The FMT* Algorithm



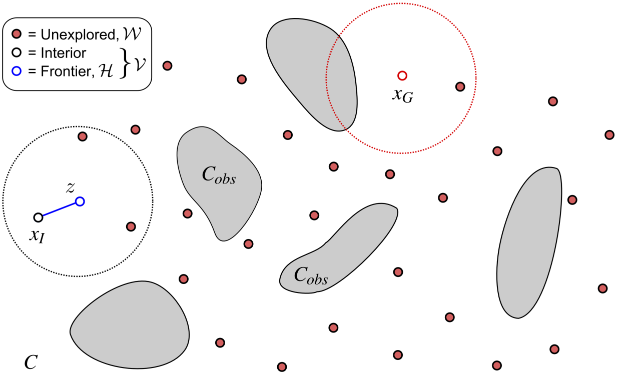
The FMT* Algorithm



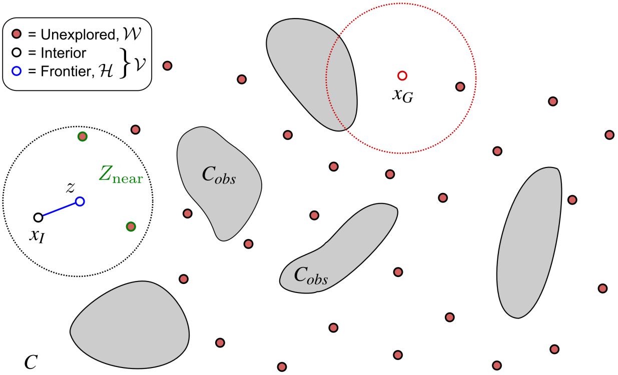
The FMT* Algorithm



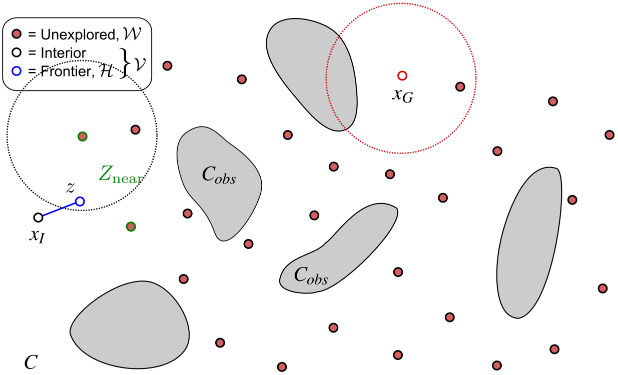
The FMT* Algorithm



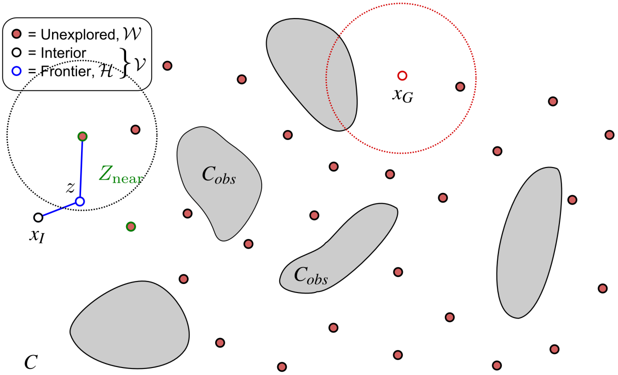
The FMT* Algorithm



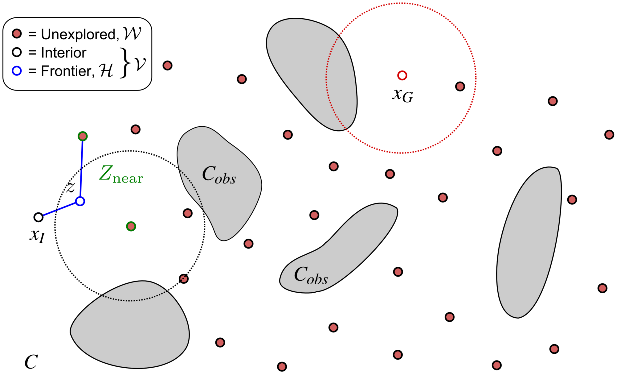
The FMT* Algorithm



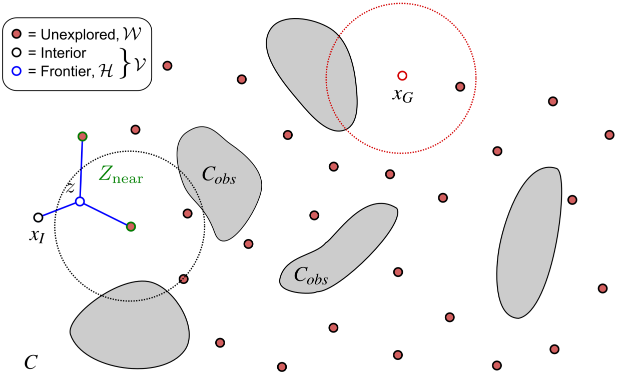
The FMT* Algorithm



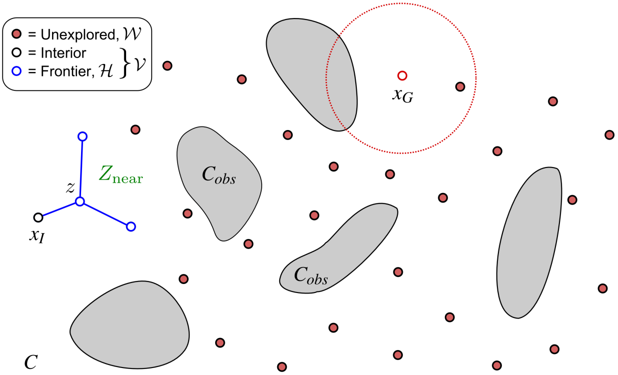
The FMT* Algorithm



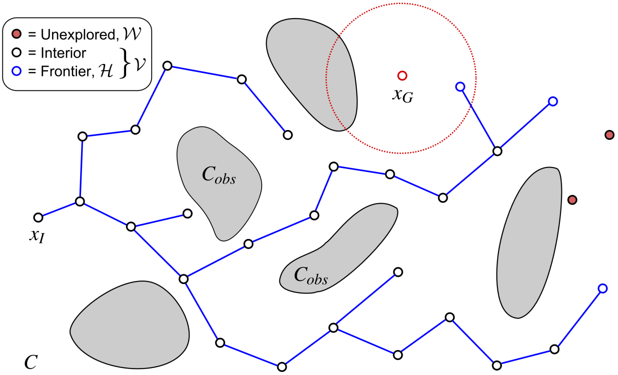
The FMT* Algorithm



The FMT* Algorithm



The FMT* Algorithm



The FMT* Algorithm